Enhancing the Comprehension of Signal Processing Principles using Audio Exercises with MATLAB*

J.W. Pierre, R.F. Kubichek, and J.C. Hamann

Department of Electrical Engineering
University of Wyoming
Laramie, WY 82071, USA

Abstract

Courses in digital signal processing (DSP) and linear systems are frequently viewed by students as abstract and impractical due to heavy emphasis on mathematical models. Unlike more physical topics such as electronics and microprocessors, hands-on experiences are difficult to achieve using standard "pencil and paper" homework assignments. Such courses can benefit greatly by applying a variety of teaching styles to help a wider range of students who have different learning styles. In particular, DSP courses can be made less abstract by employing homework exercises and classroom demonstrations that employ multimedia technologies. This paper describes a number of audio signal processing homework exercises used to reinforce concepts of signal processing. These exercises include some fundamental concepts of DSP (quantization, aliasing, Fourier analysis, and filtering) and more advanced areas (sampling rate conversion, LCMV filtering, adaptive filtering, and speech processing). All these exercises use the signal processing and audio capabilities of MATLAB. A web page for these homework exercises is being developed at wwweng.uwyo.edu/electrical/dsp_audio.

I. Introduction

Despite the instructor's best efforts, many students complete required signals and systems classes feeling that the field has little or no practical application. This is not unlike typical student reaction to electromagnetics courses where triple integrals and vector mathematics often obscure its fundamental importance. In contrast to student perceptions, signal processing technology is extensively used in a wide variety of applications including communications, control, and is especially predominant in modern consumer electronics. Getting the student to make the connection between the abstract mathematics and these real-world applications is critical in helping them learn and retain this information.

One approach to this problem begins by making classroom lectures address different learning styles. Some excellent work in the engineering education literature[6] discusses different

*This research was partially supported by a grant from the Center for Teaching Excellence (CTE) at the University of Wyoming.

learning styles and teaching with a variety of approaches. Kolb's learning cycle[3,6] contains the following four stages: 1) *Concrete Experience* (feeling and experiencing), 2) *Reflective Observation* (watching), 3) *Abstract Conceptualization* (thinking), and 4) *Active Experimentation* (doing). Kolb goes on to state that students typically report themselves as being one of four types of learners - *convergers*, *assimilators*, *divergers*, or *accomodators*. *Convergers* enjoy the practical application of principles and learn best from *active experimentation* and *abstract conceptualization*.

Assimilators are usually more interested in the logical basis of an idea than its practical application. They learn mostly by *abstract conceptualization* and *reflective observation*. *Divergers* tend to be creative and to generate many alternatives and they learn most easily from *concrete experience* and *reflective observation*. *Accommodators* are goal oriented, decisive, and act on feelings more than logic. Their preferred learning style is *concrete experience* and *active experimentation*. Using a variety of teaching techniques not only will result in helping a wider range of students who have different learning styles, but will also enhance student skills in their weaker areas. Furthermore, there is evidence[6] that a variety of learning activities greatly improves the long term retention of the concepts.

Engineers may be classified into any of the four categories, but the highest percentage of engineers is *convergers* and the second highest is *assimilators*. Engineering teaching styles tend to be best suited toward *assimilators* because of the emphasis on knowing the theory. *Convergers* on the other hand will most likely not learn very effectively or work very hard unless they see the application. Laboratories are an excellent way of having students do *active experimentation*, but unless the laboratory exercises are well written, the application may still be hidden from the student. Many signal processing courses are complimented by a laboratory and a number of excellent signal processing laboratory books[1,2,4,5,7,8] have been written. Yet not all DSP courses have a laboratory, and there is little discussion in the literature of homework exercises that reinforce *active experimentation* and *concrete experience*.

Here we propose a number of audio signal homework exercises for signal processing which involve *active experimentation* and *concrete experiences*. Most of today's students relate well to audio electronic applications such as CD, audio cassette tape, and multimedia computer games. By basing a variety of simple homework assignments on the sound capability found in most PCs, we can leverage off of the student's established interest in these technologies to build excitement in the signal processing aspects inherent in most of these devices. Availability of MATLAB provides an excellent environment for combining the sound capabilities of the computer with powerful graphing and analysis tools being studied in the classroom. These homework assignments are intentionally less involved than full laboratory exercises which allows the student time to actively experiment with the data. Most importantly, they acquire concrete experiences that are crucial for maximum understanding and retention of knowledge. In the laboratory book[7], a number of excellent audio laboratories are presented. In an introductory DSP or discrete signals and systems course, a student learning about sampling and aliasing will appreciate the concepts more, if they can hear what aliasing of a speech signal sounds like. In a more advanced course such as adaptive filtering, the concept of an adaptive filter can be made more tangible by designing a filter which removes a time varying interference.

The next section describes some MATLAB audio related exercises that reinforce many of the fundamental concepts of DSP, including quantization, aliasing, Fourier analysis, and filtering. The following section contains more advanced sound exercises including sampling rate conversion, LCMV filtering, adaptive filtering, and speech processing. All of these exercises are intended to be no more complicated than the average homework problem and to reinforce the concepts discussed in class. A web page for all of these exercises is being developed at wwweng.uwyo.edu/electrical/dsp_audio. The final section concludes the paper.

II. Reinforcing Fundamental DSP Concepts with Audio Exercises

In this section three audio homework problems are described that are appropriate for a discrete signals and systems course or an introductory DSP course. These exercises use MATLAB's sound output capabilities to illustrate the concepts of aliasing, quantization, Fourier analysis, and filtering. Most engineering homework assignments are very quantitative in nature. These assignments tend to be more qualitative because of the nature of listening to sound, but the students are instructed to answer the questions as quantitatively as possible. An assumption is that students have already been exposed to MATLAB. If not, a tutorial exercise and handouts may be provided to the students.

II.a. Aliasing and Sampling

The phrase "sampling theorem" contains the magic word "theorem" which many students learn to disdain. When many undergraduates hear this word, it is their cue to go to sleep, because to them a "theorem" is just mathematical abstraction. This exercise tries to remove that abstraction by giving the students a *concrete experience* with sampling and aliasing of audio signals. This exercise has two parts. The first examines sampling and aliasing of a sinusoidal signal, and the second looks at sampling and aliasing of a speech signal. A beneficial aspect of working with a sinusoid is that the students can plot the sampled signals and see that aliasing has occurred as well as listening to the aliasing. Yet, for many students, this is too artificial, and for it to be a *concrete experience* they need to hear aliasing of an actual speech signal.

In this exercise the student generates a 2400 Hz sinusoid. For each part of the problem the student must calculate if aliasing will occur and if so to what frequency the sinusoid will alias. Five thousandths of a second of data is used for plotting, while one second of data is used for the audio output. The MATLAB script file for the audio portion is shown in Figure 1. Initially, the sampling rate is 36000 samples/s. The data is plotted and the sound played. Next, the sinusoid is sampled at 18000 samples/s. The two sampled signals are plotted on top of each other to see that no aliasing has occurred. The signal is played to see if any audible difference can be heard. Finally, the sinusoid is sampled at 3000 samples/s, and plotted in comparison to the signal sampled at 36000 samples/s. The 2400 Hz tone has been aliased to 600 Hz. The signal is played, and the students can hear that the tone sounds much lower in frequency, because of the aliasing.

In the second half of this exercise, the students work with a speech file which was initially sampled at 11,025 samples/s. The speech is played back at the original sampling rate. Next, every other point is discarded in the sampled speech signal, reducing the sampling rate by a factor of two. This speech is played back at the new sampling rate, and the students are asked if any audible aliasing has occurred. The signal is then down sampled by another factor of two and the same question asked. Many students will continue to decrease the sampling rate by various factors out of curiosity for how it will sound. Although this exercise is very basic, it has consistently helped students in their understanding and retention of sampling and aliasing, especially students needing *active experimentation* and *concrete experience*.

```
Fo=2400;     %frequency,
Fs=36000; T=1/Fs;  %sampling rate and period
t=0:T:1;        %time vector for one second of data
x=cos(2*pi*Fo*t);   %signal
sound(x,Fs), pause   %play back at 36 ksamples/s
Fs1=18000; T1=1/Fs1; %new sampling rate and period
t1=0:T1:1;    %time vector for one second of data
x1=cos(2*pi*Fo*t1);  %signal
sound(x1,Fs1), pause   %play back at 18 ksamples/s
Fs2=3000; T2=1/Fs2; %new sampling rate and period
t2=0:T2:1;     %time vector for one second of data
x2=cos(2*pi*Fo*t2);   %signal
sound(x2,Fs2)   %play back at 3 ksamples/s
```

Figure 1. MATLAB Script for audio portion of the sampling of a sinusoid exercise.

II.b. Quantization

Quantization is a concept that students usually understand fairly well, but where they have difficulty is in obtaining insight into the significance of the quantization error. They easily grasp the fact that for each additional bit, there is a 6 dB higher signal to quantization noise ratio. What they struggle with is why that may be important. "Hearing" the effect of quantization on a speech signal can be very constructive for students.

In this exercise the students listen to a speech signal quantized by different numbers of bits. First the students listen to the original sampled speech signal quantized to 8 bits. They then quantize the signal to only 4 bits using MATLAB, listen to the changes in the sound and try to "hear" the quantization noise. They also plot a portion of the speech signal at both 8 and 4 bits on the same plot. It is also insightful to listen to the error (i.e. the difference between the 8 bit speech and the 4 bit speech). This error is also plotted for a portion of the speech signal. Other levels of quantization may be used. A similar exercise is part of an excellent set of quantization exercises contained in a book[2] on computer exercises for signal processing.

II.c. Fourier Analysis and Notch Filtering

An active exercise, which helps students with a conceptual understanding of Fourier analysis and filtering, involves the corruption of a speech signal by a sinusoidal tone. The assignment can be divided into two parts. The first involves the Fourier analysis of the signal to identify the frequency of the tone, and the second involves filtering to remove the tone. Each part can be one homework problem in separate assignments.

Students consistently struggle with Fourier analysis because of the mathematical nature of the problem and the difficulty in conceptualizing applications. In this exercise the students are first asked to listen to the noisy speech and approximately identify the signal that is corrupting the speech signal. The speech can be heard but is clearly dominated by an obnoxious sinusoidal tone. A portion of the corrupted speech file is plotted to see what it looks like in the time domain. The students are then asked to identify the frequency of the tone by computing the DFT of a portion of the signal and plotting the magnitude of the DFT. This is not an elaborate exercise, but the students see the relevance of Fourier analysis.

A notch filter is one of the more intuitive filters a student can first be introduced to for a number of reasons. First, a basic notch is a simple FIR filter with only three coefficients. Second, the design of a notch helps students to better understand the relationship between the location of the zeros/poles in the z-plane and the frequency response of a filter. Third, this exercise clearly demonstrates the application of a filter to remove an undesired component of a signal. Being able to hear this effect has greatly reinforced and motivated student understanding of digital filtering. This is an ideal exercise after the students have been exposed to difference equations, system functions, and frequency response. If an instructor wants a homework exercise in discrete signals and systems, which clearly demonstrates digital filters and motivates students to take a higher level course in digital filtering and signal processing, this is an effective assignment.

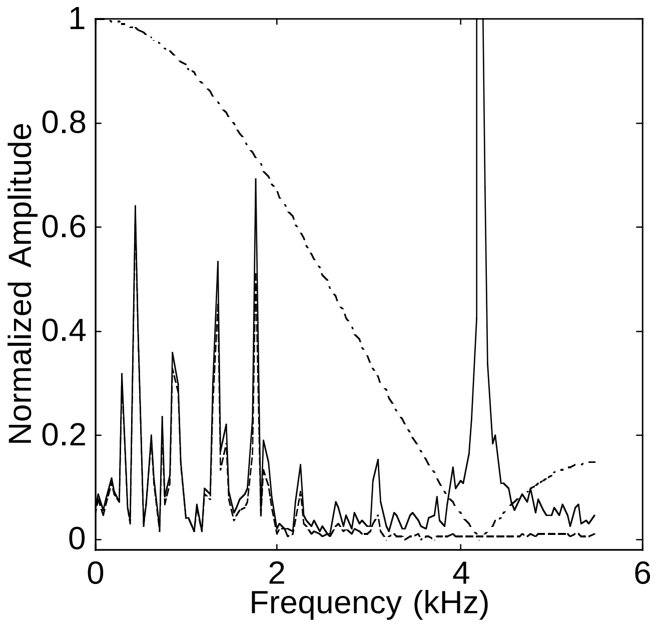

Figure 2. Fourier Analysis and Notch Filtering (dash dot - filter frequency response, solid - normalized magnitude of DFT of original speech, dashed - normalized magnitude of DFT of filtered speech).

The assignment asks the students to design an FIR notch filter to remove the tone identified in the earlier assignment. An IIR filter may also be assigned, but the simplicity of the FIR notch filter is especially nice in this first assignment in digital filtering. First, the corrupted speech is played back and a portion of the speech is plotted. A filter is then designed and MATLAB's convolve ("conv") command is used to compute the filter output. The filtered speech is then played and comments are made on what was heard and why. Students can also be asked to plot the frequency response of the filter and to recompute the DFT of the filtered signal, which helps them visualize the effectiveness of the filter at removing the sinusoidal tone, see Figure 2.

III. Reinforcing Advanced DSP Concepts with Audio Exercises

III.a. Sampling Rate Conversion

The fundamentals of interpolation and decimation are becoming more frequently taught in a senior-level digital signal processing course. Listening to decimated or interpolated speech allows students to see that the sampling rate may be digitally altered while the content of the signal is left essentially intact. Integer decimation, interpolation, or a combination of the two may be assigned. Below, an integer decimation exercise is described. This exercise can be related back nicely to the exercise on aliasing. That is, the speech output of a properly filtered and down sampled decimator can be compared with directly down sampling the data, where aliasing can be heard.

This assignment asks the students to design a decimator to reduce the sampling rate of a speech signal by a factor of two from 8 ksamples/s to 4 ksamples/s. First the students listen to the speech and then they design a decimator system including filtering. The speech is then decimated and listened to and they discuss how the output speech compares with the original. Additional parts may be added to the assignment. The students may be asked to further decimate the data until clear differences can be heard in the audio quality of the speech. They may also be asked to interpolate the speech back to the original sampling rate and compare how they sound. They can also plot the magnitude of the DFT before and after the decimation.

III.b. MVDR Filter

A topic frequently covered in an advanced signal processing course is Linear Constrained Minimum Variance (LCMV) filtering. The simplest LCMV filter is a Minimum Variance Distortionless Response (MVDR) filter where the gain is constrained to unity at a desired frequency while the filter coefficients are chosen to minimize the filter output power. A homework exercise, which uses an MVDR filter to remove a tone at an unknown frequency from a speech signal, is described below. In a way, this is a notch filter where the notch frequency is not known apriori. In fact the same speech data that was used in the notch filter exercise described in section II.c. may be used for this exercise.

In this exercise the students design a three tap MVDR filter. First, the students listen to and plot a portion of the speech. In designing the MVDR filter, DC is chosen for the constraint of unity gain, so that the primary low frequency content of the speech is preserved. Certainly more elaborate LCMV designs could be made, but this design keeps the exercise reasonable in scope for one homework problem in an assignment and reinforces the concepts underlying LCMV filters. Since the correlation matrix needed for the filter design is unknown, the students estimate it from the corrupted speech signal. After the filter coefficients are computed, the students filter the data and listen to the output. They also plot the frequency response of the filter to see that it is a notch filter. Of course for this to work, the tone needs to be the dominant portion of the speech signal so that the three tap MVDR filter focuses on minimizing the power from the tone. Also, the frequency of the tone should not be too low.

III.c. Generalized Sidelobe Cancellor

This problem implements the MVDR filter using the generalized sidelobe cancellor structure to separate the constraints from the minimization. The objective is to design an MVDR filter which will adapt to changing conditions. The length of the MVDR filter is three. The generalized sidelobe cancellor structure, shown in Figure 3, is to be used to implement the filter.

In this exercise, an adaptive LMS filter is used in conjunction with a generalized sidelobe cancellor structure to remove an interfering tone, with time varying frequency, from a speech signal. In listening to the speech the students can hear the dominant tone moving from a lower to a higher frequency. The blocking matrix, **B**, and filter, $\mathbf{w}_a$, are specified in the problem. The students write a MATLAB program to implement the LMS filter as the filter $\mathbf{w}_b$ and to process the signal using the generalized sidelobe cancellor. They need to choose a value for the LMS step parameter μ. The output e(n) should correspond to the desired adaptive MVDR filter output. They listen to the output, e(n), and discuss how the speech sounds. They plot the adaptive weights as a function of time. They also need to comment on the choice of $\mathbf{w}_a$ and **B**, and comment on the overall performance of the generalized sidelobe cancellor. The students appreciate applying the LMS algorithm to a problem where they can see and hear the results and the adaptive nature. The students are also asked to discuss what would happen if the interfering tone's signal strength were "weak" (relative to the speech signal strength) yet audibly annoying, and how would the performance of the generalized sidelobe cancellor change .

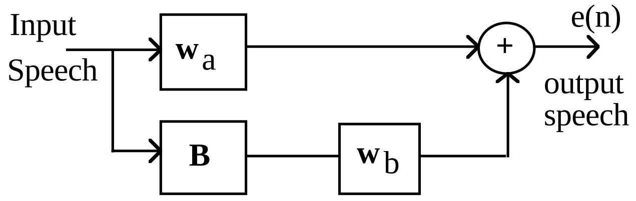

Figure 3. Generalized Sidelobe Cancellor

III.d. Companded PCM

In this problem, students can experiment with basic speech coding approaches, and develop performance curves similar to the ones in their textbooks. This leads to much deeper understanding of the material.

Pulse-code modulation (PCM) is the most direct method for digital speech transmission. When combined with logarithmic compression of the speech amplitude (called "companding"), high quality speech can be transmitted using only 8 bits/sample. This experiment compares the performance of linear PCM with mu-255 law companding, which is the standard for digital voice in the North American telephone system.

For efficient coding, the ratio of signal power to quantization noise power should be approximately constant over the full range of signal levels. Companding achieves this using logarithmic compression of the waveform so that low amplitudes are quantized using finer step sizes than high amplitudes. In contrast, linear PCM digitizes both high and low level signals using equal step size which produces constant noise power for all signal levels.

The students write MATLAB code to compute linear and companded PCM speech for 8 bit resolution, and plot SNR vs. dynamic range for both methods. Their results compare well with published results, and this motivates the student to experiment with different resolutions and speech content. More details on this exercise can be found in another paper[9].

IV. Conclusions

In the near future, more multimedia techniques will be used in signal processing education. This paper described audio exercises designed to be used as homework problems to reinforce students conceptual understanding of signal processing. Previous engineering education literature[6] emphasizes the pedagogical advantages of teaching with a variety of techniques. The audio homework exercises in this manuscript give students some of the needed *active experimentation* and *concrete experiences* to more fully understand and motivate the signal processing theory.

Acknowledgement

The authors would like to thank Dr. Jim Stice for his insights in engineering education, which he shared while on a one year appointment as the H.T. Pearson Chair in Engineering at the University of Wyoming.

Bibliography

1. Alkin O., *Digital Signal Processing a Laboratory Approach Using PC-DSP*, Prentice Hall, Englewood Cliffs, N.J., 1993.
2. McClellan J.H., Burrus C.S., Oppenheim A.V., Parks T.W., Schafer R.W., and Schuessler H.W., *Computer-Based Exercises for Signal Processing Using MATLAB 5*, Prentice Hall, Englewood Cliffs, N.J., 1998.
3. Kolb D.A., *Experiential Learning: Experience as the Source of Learning and Development*, Prentice-Hall, Englewood Cliffs, N.J., 1984.
4. Smith M.J.T. and Mersereau R.M., *Introduction to Digital Signal Processing A Computer Laboratory Textbook*, John Wiley & Sons, Inc., New York, 1992.
5. Smith M.J.T. and Mersereau R.M., *Digital Filtering A Computer Laboratory Textbook*, John Wiley & Sons, Inc., New York, 1994.
6. Stice J.E., "Using Kolb's Learning Cycle to Improve Student Learning," *Engineering Education*, pp. 291-296, February 1987.
7. Stonick V. and Bradley K., *Labs for Signals and Systems Using MATLAB*, PWS Publishing Company, Boston, 1996.
8. Buck, J.R., Daniel M.M., and Singer A.C., *Computer Explorations in Signals and Systems*, Prentice Hall, Englewood Cliffs, N.J., 1997.

9. Kubichek, R.F., “Using Matlab in a Speech Signal Processing Class,” *Proceedings of ASEE Annual Conference*, pp. 1207-1210, June 1994.

JOHN W. PIERRE
John Pierre received the BS degree in electrical engineering with a minor in economics from Montana State University in 1986. He also received the MS degree in electrical engineering with a minor in statistics and the Ph.D. degree in electrical engineering from the University of Minnesota in 1989 and 1991, respectively. He worked as an electrical design engineer at Tektronix before attending the University of Minnesota. Since 1992, he has been on the faculty in the Electrical Engineering Department at the University of Wyoming, where he is currently an associate professor. His research interests include signal processing, sensor array processing, system identification, and signal processing education.

ROBERT F. KUBICHEK
Robert Kubichek received BS degrees in EE and computer science in 1976, and the MS EE in 1977 from the University of Wyoming. He worked for Boeing Computer Services until 1979 developing CAD/CAM software, and returned to the University of Wyoming to complete a Ph.D. in electrical engineering applying pattern recognition methods to seismic oil exploration. He worked for the BDM Corporation on communications modeling problems, and then for the U.S. Department of Commerce Institute for Telecommunications Sciences where he researched speech communications system performance measures. He is currently an associate professor at the University of Wyoming where he has been on the faculty since Fall 1991.

JERRY C. HAMANN
Prior to completing the PhD in Electrical Engineering at the University of Wisconsin-Madison, Jerry Hamann received BS and MS degrees from the University of Wyoming and worked as a Product Support Engineer for Data Acquisition products at Hewlett-Packard in Loveland, Colorado. Dr. Hamann is currently an Assistant Professor in the Department of Electrical Engineering at the University of Wyoming. His research interests include instrumentation and signal processing, with applications in automatic control and distributed intelligent systems. He is a member of IEEE, ASEE, and IMAPS.